Pandas Library Basics

Lec:4 Series, Dataframe and CSV(comma Seprated Value)

In [80]:

```
import pandas as pd
```

In [81]:

```
#2 main datatypes
#series 1D and dataframes 2D

series= pd.Series(["BMW","Toyota", "Honda"])
```

In [82]:

```
series
```

Out[82]:

```
0       BMW
1    Toyota
2     Honda
dtype: object
```

In [83]:

```
#series =1-dimensional
```

In [84]:

```
colours= pd.Series(["Red", "blue", "white"])
colours
```

Out[84]:

```
0      Red
1     blue
2    white
dtype: object
```

In [85]:

```
#series is less common and one dimensional data type supported by python
#but Data frames are two dimensional data types and are commonly use in ML projects
```

In [86]:

```
#Dataframe = 2-Dimensional

#it take python dictionaries we can combine to series data also in dataframe

car_data= pd.DataFrame({"Car_maker":series, "colors":colours})

#for printing data frame simply write name of it Ex-

car_data
```

Out[86]:

	Car_maker	colors
0	BMW	Red
1	Toyota	blue
2	Honda	white

In [87]:

```
#Creating data frames from scratch is little tedious so we're gonna improt data from intern
```

In [88]:

```
#first save data as csv

car_sales= pd.read_csv("car-sales.csv")
```

In [89]:

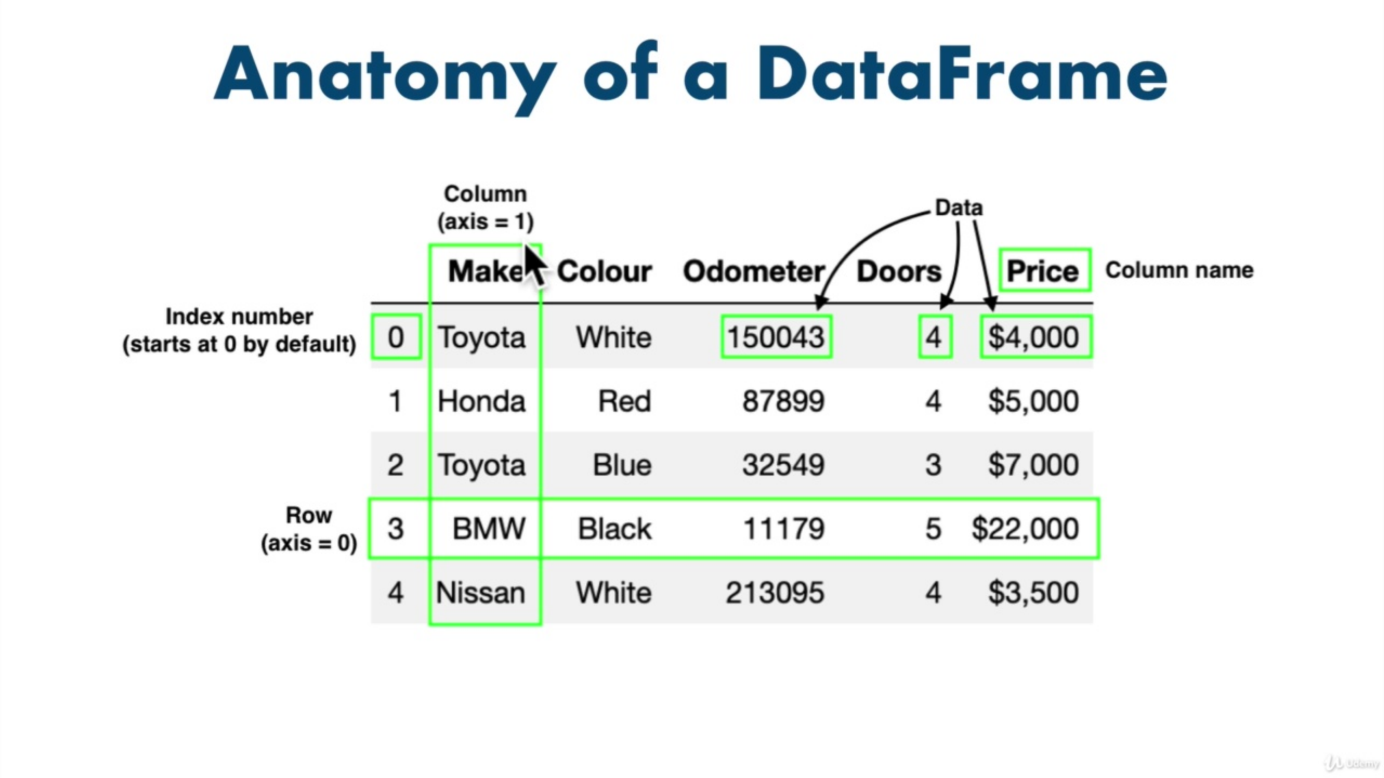
```
#print data frame

car_sales
```

Out[89]:

```
     Make  Colour  Odometer (KM)  Doors       Price
0  Toyota   White         150043      4   $4,000.00
1   Honda     Red          87899      4   $5,000.00
2  Toyota    Blue          32549      3   $7,000.00
3     BMW   Black          11179      5  $22,000.00
4  Nissan   White         213095      4   $3,500.00
5  Toyota   Green          99213      4   $4,500.00
6   Honda    Blue          45698      4   $7,500.00
7   Honda    Blue          54738      4   $7,000.00
8  Toyota   White          60000      4   $6,250.00
9  Nissan   White          31600      4   $9,700.00
```

Anatomy of Data Frame



Anatomy of a DataFrame

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043	4	$4,000
1	Honda	Red	87899	4	$5,000
2	Toyota	Blue	32549	3	$7,000
3	BMW	Black	11179	5	$22,000
4	Nissan	White	213095	4	$3,500

In [90]:

```
#Exporting a dataframe
#index = false remove id**
car_sales.to_csv("exported-car-sales.csv", index=False)
```

In [91]:

```
# reading exported Data

export_car_sales=pd.read_csv("exported-car-sales.csv")

export_car_sales
```

Out[91]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00
5	Toyota	Green	99213	4	$4,500.00
6	Honda	Blue	45698	4	$7,500.00
7	Honda	Blue	54738	4	$7,000.00
8	Toyota	White	60000	4	$6,250.00
9	Nissan	White	31600	4	$9,700.00

In [92]:

```
#notice also included some unnamed columns to remove this you need to export index= false
```

Lec :6 Describing Data with Pandas

In [93]:

```
# Attributes --> describe meta information about car_sales

# type of data

car_sales.dtypes

# Function --> steps
```

Out[93]:

```
Make              object
Colour            object
Odometer (KM)      int64
Doors              int64
Price             object
dtype: object
```

In [94]:

```
# return list of columns

car_sales.columns
```

Out[94]:

```
Index(['Make', 'Colour', 'Odometer (KM)', 'Doors', 'Price'], dtype='object')
```

In [95]:

```
# assigning columns to other variable for purpose of manipulation
car_coloumns= car_sales.columns

#printing car_coloumns
car_coloumns
```

Out[95]:

```
Index(['Make', 'Colour', 'Odometer (KM)', 'Doors', 'Price'], dtype='object')
```

In [96]:

```
# index range
car_sales.index
```

Out[96]:

```
RangeIndex(start=0, stop=10, step=1)
```

In [97]:

```
# Functions --> performs some information

#describe Function--> it show some satistical information about data only interger
# varibles

car_sales.describe()
```

Out[97]:

	Odometer (KM)	Doors
count	10.000000	10.000000
mean	78601.400000	4.000000
std	61983.471735	0.471405
min	11179.000000	3.000000
25%	35836.250000	4.000000
50%	57369.000000	4.000000
75%	96384.500000	4.000000
max	213095.000000	5.000000

In [98]:

```
# info function

car_sales.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   Make           10 non-null     object
 1   Colour         10 non-null     object
 2   Odometer (KM)  10 non-null     int64 
 3   Doors          10 non-null     int64 
 4   Price          10 non-null     object
dtypes: int64(2), object(3)
memory usage: 528.0+ bytes
```

In [99]:

```
# statictical anlysis of data
#ex-- mean
car_sales.mean()
```

```
C:\Users\ay569\AppData\Local\Temp\ipykernel_703224\3146424867.py:3: FutureWa
rning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_o
nly=None') is deprecated; in a future version this will raise TypeError.  Se
lect only valid columns before calling the reduction.
  car_sales.mean()
```

Out[99]:

```
Odometer (KM)    78601.4
Doors                4.0
dtype: float64
```

In [100]:

```
# mean on individual series

car_prices= pd.Series([3000, 1220,141421])
car_prices.mean()
```

Out[100]:

```
48547.0
```

In [101]:

```
# sum of all numerical coulumns
car_sales.sum()
```

Out[101]:

```
Make                ToyotaHondaToyotaBMWNissanToyotaHondaHondaToyo...
Colour                  WhiteRedBlueBlackWhiteGreenBlueBlueWhiteWhite
Odometer (KM)                                                  786014
Doors                                                              40
Price               $4,000.00$5,000.00$7,000.00$22,000.00$3,500.00...
dtype: object
```

In [102]:

```
# select single coulumn

car_sales["Doors"].sum()
```

Out[102]:

```
40
```

In [103]:

```
# lenght of row
len(car_sales)
```

Out[103]:

```
10
```

Lec 7:Selecting and Viewing Data with Pandas

In [104]:

```
# head return top five rows of your data

car_sales.head()
```

Out[104]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00

In [105]:

```
# head for desired row
car_sales.head(10)
```

Out[105]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00
5	Toyota	Green	99213	4	$4,500.00
6	Honda	Blue	45698	4	$7,500.00
7	Honda	Blue	54738	4	$7,000.00
8	Toyota	White	60000	4	$6,250.00
9	Nissan	White	31600	4	$9,700.00

In [106]:

```
# to look at bottom of your dataframe

car_sales.tail()
car_sales.tail(10)
```

Out[106]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00
5	Toyota	Green	99213	4	$4,500.00
6	Honda	Blue	45698	4	$7,500.00
7	Honda	Blue	54738	4	$7,000.00
8	Toyota	White	60000	4	$6,250.00
9	Nissan	White	31600	4	$9,700.00

In [107]:

```
# .loc & .iloc
animals= pd.Series(["cat","dog","bird","panda", "snake"])
animals
```

Out[107]:

```
0      cat
1      dog
2     bird
3    panda
4    snake
dtype: object
```

In [108]:

```
#for manual indexing

animals= pd.Series(["cat","dog","bird","panda", "snake"],index=[0,3,9,8,3])

animals
```

Out[108]:

```
0      cat
3      dog
9     bird
8    panda
3    snake
dtype: object
```

In [109]:

```
# .loc-- stands for loaction used for printing item on particular location
# it refers to index
animals.loc[3]
```

Out[109]:

```
3      dog
3    snake
dtype: object
```

In [110]:

```
# more example--> loc
car_sales.loc[3]
```

Out[110]:

```
Make                     BMW
Colour                 Black
Odometer (KM)          11179
Doors                      5
Price             $22,000.00
Name: 3, dtype: object
```

In [111]:

```
# .iloc --> refers to position
animals.iloc[3]
```

Out[111]:

```
'panda'
```

In [112]:

```
car_sales.iloc[3]
```

Out[112]:

```
Make                      BMW
Colour                  Black
Odometer (KM)           11179
Doors                       5
Price              $22,000.00
Name: 3, dtype: object
```

In [113]:

```
# loc and iloc also in helps in slicing

animals.iloc[:3]
```

Out[113]:

```
0     cat
3     dog
9    bird
dtype: object
```

In [114]:

```
car_sales.iloc[:3]
```

Out[114]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00

In [115]:

```
# printing particular column

car_sales["Make"]
```

Out[115]:

```
0    Toyota
1     Honda
2    Toyota
3       BMW
4    Nissan
5    Toyota
6     Honda
7     Honda
8    Toyota
9    Nissan
Name: Make, dtype: object
```

In [116]:

```
car_sales["Colour"]
```

Out[116]:

```
0    White
1      Red
2     Blue
3    Black
4    White
5    Green
6     Blue
7     Blue
8    White
9    White
Name: Colour, dtype: object
```

In [117]:

```
# other way to selecting cloumn
# esier to type

# ***** Note: if your column name have space it dosen't work

car_sales.Make
```

Out[117]:

```
0    Toyota
1     Honda
2    Toyota
3       BMW
4    Nissan
5    Toyota
6     Honda
7     Honda
8    Toyota
9    Nissan
Name: Make, dtype: object
```

In [118]:

```
# select single column with some conditions
# ***

car_sales[car_sales["Make"]== "Toyota"]
```

Out[118]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
2	Toyota	Blue	32549	3	$7,000.00
5	Toyota	Green	99213	4	$4,500.00
8	Toyota	White	60000	4	$6,250.00

In [119]:

```
# Ex-2

car_sales[car_sales["Odometer (KM)"]>10000]
```

Out[119]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00
5	Toyota	Green	99213	4	$4,500.00
6	Honda	Blue	45698	4	$7,500.00
7	Honda	Blue	54738	4	$7,000.00
8	Toyota	White	60000	4	$6,250.00
9	Nissan	White	31600	4	$9,700.00

8. Selecting and Viewing Data with Pandas Part 2

In [120]:

```
# cross over of two columns

pd.crosstab(car_sales["Make"], car_sales["Doors"])
```

Out[120]:

Doors	**3**	**4**	**5**
Make			
BMW	0	0	1
Honda	0	3	0
Nissan	0	2	0
Toyota	1	3	0

In [121]:

```
# Groupby--> group the dataframe by cloumn and apply some operation

car_sales.groupby(["Make"]).mean()
```

Out[121]:

	Odometer (KM)	Doors
Make		
BMW	11179.000000	5.00
Honda	62778.333333	4.00
Nissan	122347.500000	4.00
Toyota	85451.250000	3.75

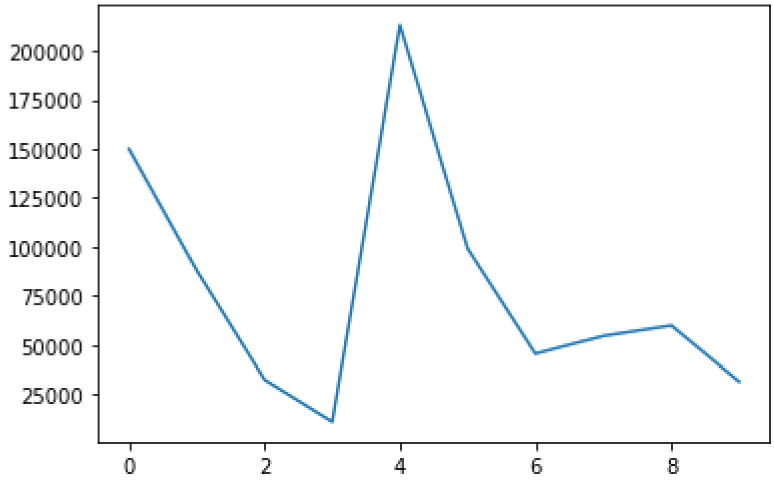
In [122]:

```
## ploting data

car_sales["Odometer (KM)"].plot()
```

Out[122]:

```
<AxesSubplot:>
```



In [123]:

```
# if plot doesn't show up write this

# %matplotlib inline
# import matplotlib.pyplot as plt
```

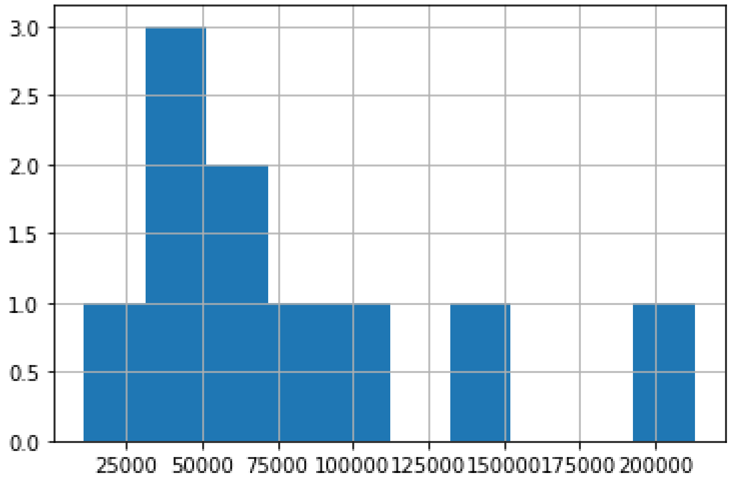
In [124]:

```
#plotting histogram

car_sales["Odometer (KM)"].hist()
```

Out[124]:

```
<AxesSubplot:>
```



In [125]:

```
car_sales["Price"].dtype
```

Out[125]:

```
dtype('O')
```

In [126]:

```
# conveting object to integer

car_sales["Price"]= car_sales["Price"]. str.replace('[\$\,\.]','').astype(int)
```

```
C:\Users\ay569\AppData\Local\Temp\ipykernel_703224\2316547196.py:3: FutureWa
rning: The default value of regex will change from True to False in a future
version.
  car_sales["Price"]= car_sales["Price"]. str.replace('[\$\,\.]','').astype
(int)
```

In [127]:

```
car_sales
```

Out[127]:

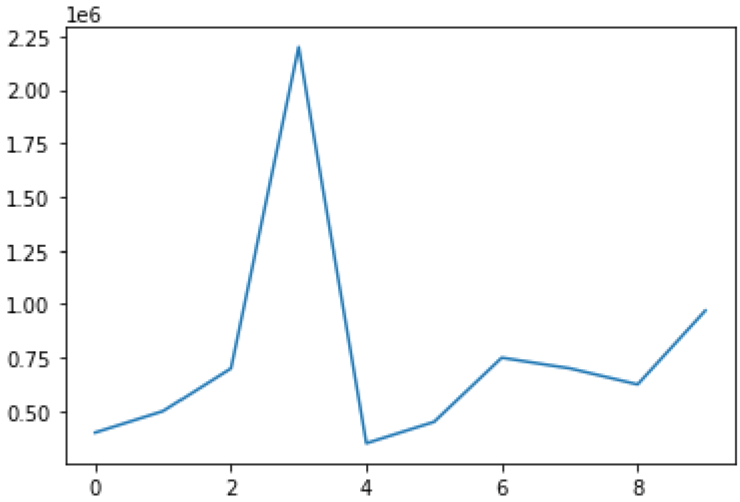
	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	400000
1	Honda	Red	87899	4	500000
2	Toyota	Blue	32549	3	700000
3	BMW	Black	11179	5	2200000
4	Nissan	White	213095	4	350000
5	Toyota	Green	99213	4	450000
6	Honda	Blue	45698	4	750000
7	Honda	Blue	54738	4	700000
8	Toyota	White	60000	4	625000
9	Nissan	White	31600	4	970000

In [128]:

```
car_sales["Price"].plot()
```

Out[128]:

```
<AxesSubplot:>
```



Lec 9. Manipulating Data

In [129]:

```
# convert the string in lower case

car_sales["Make"].str.lower()
```

Out[129]:

```
0    toyota
1     honda
2    toyota
3       bmw
4    nissan
5    toyota
6     honda
7     honda
8    toyota
9    nissan
Name: Make, dtype: object
```

In [130]:

```
car_sales
```

Out[130]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	400000
1	Honda	Red	87899	4	500000
2	Toyota	Blue	32549	3	700000
3	BMW	Black	11179	5	2200000
4	Nissan	White	213095	4	350000
5	Toyota	Green	99213	4	450000
6	Honda	Blue	45698	4	750000
7	Honda	Blue	54738	4	700000
8	Toyota	White	60000	4	625000
9	Nissan	White	31600	4	970000

In [131]:

```
## change will dosen't save because you didn't save it

car_sales["Make"]= car_sales["Make"].str.lower()
```

In [132]:

```
car_sales
```

Out[132]:

	Make	Colour	Odometer (KM)	Doors	Price
0	toyota	White	150043	4	400000
1	honda	Red	87899	4	500000
2	toyota	Blue	32549	3	700000
3	bmw	Black	11179	5	2200000
4	nissan	White	213095	4	350000
5	toyota	Green	99213	4	450000
6	honda	Blue	45698	4	750000
7	honda	Blue	54738	4	700000
8	toyota	White	60000	4	625000
9	nissan	White	31600	4	970000

In [133]:

```
# Missing data

#importing file with missing data

car_sales_missing= pd.read_csv("car-sales-missing-data.csv")

car_sales_missing
```

Out[133]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.0	4.0	$4,000
1	Honda	Red	87899.0	4.0	$5,000
2	Toyota	Blue	NaN	3.0	$7,000
3	BMW	Black	11179.0	5.0	$22,000
4	Nissan	White	213095.0	4.0	$3,500
5	Toyota	Green	NaN	4.0	$4,500
6	Honda	NaN	NaN	4.0	$7,500
7	Honda	Blue	NaN	4.0	NaN
8	Toyota	White	60000.0	NaN	NaN
9	NaN	White	31600.0	4.0	$9,700

In [134]:

```
## NaN --> for no value
```

In [135]:

```
car_sales_missing["Odometer"].mean()
```

Out[135]:

```
92302.66666666667
```

In [136]:

```
# how to fill missing values
# fill data with mean values
car_sales_missing["Odometer"].fillna(car_sales_missing["Odometer"].mean())
```

Out[136]:

```
0    150043.000000
1     87899.000000
2     92302.666667
3     11179.000000
4    213095.000000
5     92302.666667
6     92302.666667
7     92302.666667
8     60000.000000
9     31600.000000
Name: Odometer, dtype: float64
```

In [137]:

```
car_sales_missing
```

Out[137]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.0	4.0	$4,000
1	Honda	Red	87899.0	4.0	$5,000
2	Toyota	Blue	NaN	3.0	$7,000
3	BMW	Black	11179.0	5.0	$22,000
4	Nissan	White	213095.0	4.0	$3,500
5	Toyota	Green	NaN	4.0	$4,500
6	Honda	NaN	NaN	4.0	$7,500
7	Honda	Blue	NaN	4.0	NaN
8	Toyota	White	60000.0	NaN	NaN
9	NaN	White	31600.0	4.0	$9,700

In [138]:

```
# above vaues dosen't change so assign to see changes

car_sales_missing["Odometer"]=car_sales_missing["Odometer"].fillna(car_sales_missing["Odome
```

In [139]:

```
car_sales_missing
```

Out[139]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.000000	4.0	$4,000
1	Honda	Red	87899.000000	4.0	$5,000
2	Toyota	Blue	92302.666667	3.0	$7,000
3	BMW	Black	11179.000000	5.0	$22,000
4	Nissan	White	213095.000000	4.0	$3,500
5	Toyota	Green	92302.666667	4.0	$4,500
6	Honda	NaN	92302.666667	4.0	$7,500
7	Honda	Blue	92302.666667	4.0	NaN
8	Toyota	White	60000.000000	NaN	NaN
9	NaN	White	31600.000000	4.0	$9,700

In [140]:

```
#other method inplace = true

car_sales_missing["Odometer"].fillna(car_sales_missing["Odometer"].mean(), inplace=True)
```

In [141]:

```
car_sales_missing
```

Out[141]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.000000	4.0	$4,000
1	Honda	Red	87899.000000	4.0	$5,000
2	Toyota	Blue	92302.666667	3.0	$7,000
3	BMW	Black	11179.000000	5.0	$22,000
4	Nissan	White	213095.000000	4.0	$3,500
5	Toyota	Green	92302.666667	4.0	$4,500
6	Honda	NaN	92302.666667	4.0	$7,500
7	Honda	Blue	92302.666667	4.0	NaN
8	Toyota	White	60000.000000	NaN	NaN
9	NaN	White	31600.000000	4.0	$9,700

In [142]:

```
# how to remove missing values

car_sales_missing.dropna (inplace= True)
```

In [143]:

```
car_sales_missing
```

Out[143]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.000000	4.0	$4,000
1	Honda	Red	87899.000000	4.0	$5,000
2	Toyota	Blue	92302.666667	3.0	$7,000
3	BMW	Black	11179.000000	5.0	$22,000
4	Nissan	White	213095.000000	4.0	$3,500
5	Toyota	Green	92302.666667	4.0	$4,500

In [144]:

```
# how to reaccessing droped row in of your data
# create new data frame
car_sales_missing_dropped=pd.read_csv("car-sales-missing-data.csv")
```

In [145]:

```
car_sales_missing_dropped
```

Out[145]:

	Make	Colour	Odometer	Doors	Price
0	Toyota	White	150043.0	4.0	$4,000
1	Honda	Red	87899.0	4.0	$5,000
2	Toyota	Blue	NaN	3.0	$7,000
3	BMW	Black	11179.0	5.0	$22,000
4	Nissan	White	213095.0	4.0	$3,500
5	Toyota	Green	NaN	4.0	$4,500
6	Honda	NaN	NaN	4.0	$7,500
7	Honda	Blue	NaN	4.0	NaN
8	Toyota	White	60000.0	NaN	NaN
9	NaN	White	31600.0	4.0	$9,700

In [146]:

```
# saving your dropped dataframe

car_sales_missing.to_csv("car_sales_missing_dropped")
```

Lec 10. Manipulating Data 2

In [147]:

```
# How do we crate a data from existing data
#creating new coloumn
#Cloumn from series

seats_column= pd.Series([5,5,5,5,5])

# New column called seats

car_sales["Seats"]= seats_column

car_sales
```

Out[147]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats
0	toyota	White	150043	4	400000	5.0
1	honda	Red	87899	4	500000	5.0
2	toyota	Blue	32549	3	700000	5.0
3	bmw	Black	11179	5	2200000	5.0
4	nissan	White	213095	4	350000	5.0
5	toyota	Green	99213	4	450000	NaN
6	honda	Blue	45698	4	750000	NaN
7	honda	Blue	54738	4	700000	NaN
8	toyota	White	60000	4	625000	NaN
9	nissan	White	31600	4	970000	NaN

In [148]:

```
# filling null values in column Seats

car_sales["Seats"].fillna(5, inplace=True)
```

In [149]:

```
car_sales
```

Out[149]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats
0	toyota	White	150043	4	400000	5.0
1	honda	Red	87899	4	500000	5.0
2	toyota	Blue	32549	3	700000	5.0
3	bmw	Black	11179	5	2200000	5.0
4	nissan	White	213095	4	350000	5.0
5	toyota	Green	99213	4	450000	5.0
6	honda	Blue	45698	4	750000	5.0
7	honda	Blue	54738	4	700000	5.0
8	toyota	White	60000	4	625000	5.0
9	nissan	White	31600	4	970000	5.0

In [150]:

```
# Column from Python list
# Note: in list your list size must be exact equal to your row in dataframe
fuel_economy= [9.2,2.25, 14.42, 424.2, 424.1, 24.32,21.2,32,24.24,242.4]

car_sales["Fuel per 100 KM"]= fuel_economy
```

In [151]:

```
car_sales
```

Out[151]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	Fuel per 100 KM
0	toyota	White	150043	4	400000	5.0	9.20
1	honda	Red	87899	4	500000	5.0	2.25
2	toyota	Blue	32549	3	700000	5.0	14.42
3	bmw	Black	11179	5	2200000	5.0	424.20
4	nissan	White	213095	4	350000	5.0	424.10
5	toyota	Green	99213	4	450000	5.0	24.32
6	honda	Blue	45698	4	750000	5.0	21.20
7	honda	Blue	54738	4	700000	5.0	32.00
8	toyota	White	60000	4	625000	5.0	24.24
9	nissan	White	31600	4	970000	5.0	242.40

In [162]:

```
# creating column from another column

car_sales["Total fuel used (L)"]= car_sales["Odometer (KM)"]/100* car_sales["Fuel per 100 K
car_sales
```

Out[162]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	Fuel per 100 KM	Number of Wheels	Passed road safety	Total fuel used (L)
0	toyota	White	150043	4	400000	5.0	9.20	4	True	13803.9560
1	honda	Red	87899	4	500000	5.0	2.25	4	True	1977.7275
2	toyota	Blue	32549	3	700000	5.0	14.42	4	True	4693.5658
3	bmw	Black	11179	5	2200000	5.0	424.20	4	True	47421.3180
4	nissan	White	213095	4	350000	5.0	424.10	4	True	903735.8950
5	toyota	Green	99213	4	450000	5.0	24.32	4	True	24128.6016
6	honda	Blue	45698	4	750000	5.0	21.20	4	True	9687.9760
7	honda	Blue	54738	4	700000	5.0	32.00	4	True	17516.1600
8	toyota	White	60000	4	625000	5.0	24.24	4	True	14544.0000
9	nissan	White	31600	4	970000	5.0	242.40	4	True	76598.4000

In [163]:

```
# create column from singel value

car_sales["Number of Wheels"]=4
car_sales
```

Out[163]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	Fuel per 100 KM	Number of Wheels	Passed road safety	Total fuel used (L)
0	toyota	White	150043	4	400000	5.0	9.20	4	True	13803.9560
1	honda	Red	87899	4	500000	5.0	2.25	4	True	1977.7275
2	toyota	Blue	32549	3	700000	5.0	14.42	4	True	4693.5658
3	bmw	Black	11179	5	2200000	5.0	424.20	4	True	47421.3180
4	nissan	White	213095	4	350000	5.0	424.10	4	True	903735.8950
5	toyota	Green	99213	4	450000	5.0	24.32	4	True	24128.6016
6	honda	Blue	45698	4	750000	5.0	21.20	4	True	9687.9760
7	honda	Blue	54738	4	700000	5.0	32.00	4	True	17516.1600
8	toyota	White	60000	4	625000	5.0	24.24	4	True	14544.0000
9	nissan	White	31600	4	970000	5.0	242.40	4	True	76598.4000

In [182]:

```
car_sales["Passed road safety"]= True
car_sales
```

Out[182]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	Fuel per 100 KM	Number of Wheels	Passed road safety
0	toyota	White	150043	4	400000	5.0	9.20	4	True
1	honda	Red	87899	4	500000	5.0	2.25	4	True
2	toyota	Blue	32549	3	700000	5.0	14.42	4	True
3	bmw	Black	11179	5	2200000	5.0	424.20	4	True
4	nissan	White	213095	4	350000	5.0	424.10	4	True
5	toyota	Green	99213	4	450000	5.0	24.32	4	True
6	honda	Blue	45698	4	750000	5.0	21.20	4	True
7	honda	Blue	54738	4	700000	5.0	32.00	4	True
8	toyota	White	60000	4	625000	5.0	24.24	4	True
9	nissan	White	31600	4	970000	5.0	242.40	4	True

In [183]:

```
car_sales.dtypes
```

Out[183]:

```
Make                   object
Colour                 object
Odometer (KM)           int64
Doors                   int64
Price                   int32
Seats                 float64
Fuel per 100 KM       float64
Number of Wheels        int64
Passed road safety       bool
dtype: object
```

In [184]:

```
# removing column

car_sales.drop("Total fuel used (L)", axis=1, inplace= True)

car_sales
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Input In [184], in <cell line: 3>()
      1 # removing column
----> 3 car_sales.drop("Total fuel used (L)", axis=1, inplace= True)
      5 car_sales

File ~\Downloads\mc_sample_project\env\lib\site-packages\pandas\util\_decora
tors.py:311, in deprecate_nonkeyword_arguments.<locals>.decorate.<locals>.wr
apper(*args, **kwargs)
    305 if len(args) > num_allow_args:
    306     warnings.warn(
    307         msg.format(arguments=arguments),
    308         FutureWarning,
    309         stacklevel=stacklevel,
    310     )
--> 311 return func(*args, **kwargs)

File ~\Downloads\mc_sample_project\env\lib\site-packages\pandas\core\frame.p
y:4954, in DataFrame.drop(self, labels, axis, index, columns, level, inplac
e, errors)
   4806 @deprecate_nonkeyword_arguments(version=None, allowed_args=["self",
"labels"])
   4807 def drop(
   4808     self,
   (...)
   4815     errors: str = "raise",
   4816 ):
   4817     """
   4818     Drop specified labels from rows or columns.
   4819
   (...)
   4952             weight  1.0     0.8
   4953     """
-> 4954     return super().drop(
   4955         labels=labels,
   4956         axis=axis,
   4957         index=index,
   4958         columns=columns,
   4959         level=level,
   4960         inplace=inplace,
   4961         errors=errors,
   4962     )

File ~\Downloads\mc_sample_project\env\lib\site-packages\pandas\core\generi
c.py:4267, in NDFrame.drop(self, labels, axis, index, columns, level, inplac
e, errors)
   4265 for axis, labels in axes.items():
   4266     if labels is not None:
-> 4267         obj = obj._drop_axis(labels, axis, level=level, errors=error
s)
   4269 if inplace:
   4270     self._update_inplace(obj)
```

```
File ~\Downloads\mc_sample_project\env\lib\site-packages\pandas\core\generi
c.py:4311, in NDFrame._drop_axis(self, labels, axis, level, errors, consolid
ate, only_slice)
   4309         new_axis = axis.drop(labels, level=level, errors=errors)
   4310     else:
-> 4311         new_axis = axis.drop(labels, errors=errors)
   4312     indexer = axis.get_indexer(new_axis)
   4314 # Case for non-unique axis
   4315 else:

File ~\Downloads\mc_sample_project\env\lib\site-packages\pandas\core\indexes
\base.py:6644, in Index.drop(self, labels, errors)
   6642 if mask.any():
   6643     if errors != "ignore":
-> 6644         raise KeyError(f"{list(labels[mask])} not found in axis")
   6645     indexer = indexer[~mask]
   6646 return self.delete(indexer)

KeyError: "['Total fuel used (L)'] not found in axis"
```

Lec 11. Manipulating Data 3

In [185]:

```
# shuffing the order of elements in dataframe

car_sales_shuffeled = car_sales.sample(frac=1)

car_sales_shuffeled
```

Out[185]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	Fuel per 100 KM	Number of Wheels	Passed road safety
5	toyota	Green	99213	4	450000	5.0	24.32	4	True
3	bmw	Black	11179	5	2200000	5.0	424.20	4	True
7	honda	Blue	54738	4	700000	5.0	32.00	4	True
4	nissan	White	213095	4	350000	5.0	424.10	4	True
6	honda	Blue	45698	4	750000	5.0	21.20	4	True
2	toyota	Blue	32549	3	700000	5.0	14.42	4	True
8	toyota	White	60000	4	625000	5.0	24.24	4	True
1	honda	Red	87899	4	500000	5.0	2.25	4	True
0	toyota	White	150043	4	400000	5.0	9.20	4	True
9	nissan	White	31600	4	970000	5.0	242.40	4	True

In [186]:

```
# Only select 20% of the data

car_sales_shuffeled.sample(frac=0.2)
car_sales_shuffeled
```

Out[186]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	Fuel per 100 KM	Number of Wheels	Passed road safety
5	toyota	Green	99213	4	450000	5.0	24.32	4	True
3	bmw	Black	11179	5	2200000	5.0	424.20	4	True
7	honda	Blue	54738	4	700000	5.0	32.00	4	True
4	nissan	White	213095	4	350000	5.0	424.10	4	True
6	honda	Blue	45698	4	750000	5.0	21.20	4	True
2	toyota	Blue	32549	3	700000	5.0	14.42	4	True
8	toyota	White	60000	4	625000	5.0	24.24	4	True
1	honda	Red	87899	4	500000	5.0	2.25	4	True
0	toyota	White	150043	4	400000	5.0	9.20	4	True
9	nissan	White	31600	4	970000	5.0	242.40	4	True

In [187]:

```
# How to reset the shfufled data

car_sales_shuffeled.reset_index(drop= True,inplace= True)

car_sales_shuffeled
```

Out[187]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	Fuel per 100 KM	Number of Wheels	Passed road safety
0	toyota	Green	99213	4	450000	5.0	24.32	4	True
1	bmw	Black	11179	5	2200000	5.0	424.20	4	True
2	honda	Blue	54738	4	700000	5.0	32.00	4	True
3	nissan	White	213095	4	350000	5.0	424.10	4	True
4	honda	Blue	45698	4	750000	5.0	21.20	4	True
5	toyota	Blue	32549	3	700000	5.0	14.42	4	True
6	toyota	White	60000	4	625000	5.0	24.24	4	True
7	honda	Red	87899	4	500000	5.0	2.25	4	True
8	toyota	White	150043	4	400000	5.0	9.20	4	True
9	nissan	White	31600	4	970000	5.0	242.40	4	True

In [188]:

```
# How to apply function on columns

# labda is type of function
car_sales["Odometer (KM)"]= car_sales["Odometer (KM)"]. apply(lambda x: x/1.6)

car_sales
```

Out[188]:

	Make	Colour	Odometer (KM)	Doors	Price	Seats	Fuel per 100 KM	Number of Wheels	Passed road safety
0	toyota	White	93776.875	4	400000	5.0	9.20	4	True
1	honda	Red	54936.875	4	500000	5.0	2.25	4	True
2	toyota	Blue	20343.125	3	700000	5.0	14.42	4	True
3	bmw	Black	6986.875	5	2200000	5.0	424.20	4	True
4	nissan	White	133184.375	4	350000	5.0	424.10	4	True
5	toyota	Green	62008.125	4	450000	5.0	24.32	4	True
6	honda	Blue	28561.250	4	750000	5.0	21.20	4	True
7	honda	Blue	34211.250	4	700000	5.0	32.00	4	True
8	toyota	White	37500.000	4	625000	5.0	24.24	4	True
9	nissan	White	19750.000	4	970000	5.0	242.40	4	True

The End*

```
Do All the assingment questions and revise regularly
```